Maskininlärning

Kunskapskontroll 2

Jakob Rask

EC Utbildning

Rapport

2024-03-22

Abstract

The MNIST dataset consists of data for hand-written images of digits ranging from 0 to 9. By evaluating different machine learning models on the MNIST dataset the best performer was picked out to act as our final model.

The goal was to create an application to take pictures of hand-written digits for the model to predict. Analysing the data and understanding how the pixel values affect the image output is crucial in understanding how to process new data. By processing the new data to resemble the training data we can successfully predict the new digit.

Innehållsförteckning

1 Inledning

Maskininlärning har fått en stor påverkan på vårt samhälle. Med ökad efterfrågan på automatisering och förbättrad användarupplevelse i en rad olika branscher, spelar maskininlärning en viktig roll för både effektivare processer och spännande innovationer.

Det som gör maskininlärning unikt är att vi genom modeller och data kan programmera datorer att själva lära sig av data, utan att ytterligare programmering behövs. Genom att det skapas mer och mer data blir möjligheterna fler och potentialen större för varje dag som går.

Maskininlärning kan till exempel användas inom bildigenkänning, som i denna uppgift, där ett visst bildobjekt ska tilldelas en viss klass.

1.1 Syfte

Uppgiften består av två delar, där den första delen handlar om att använda maskininlärning för att modellera indata avseende handritade siffror. Syftet är att ta fram den bästa maskininlärningsmodellen utifrån de modeller som valts ut.

Den andra delen går ut på att skapa en applikation som har en kamerafunktion, med syftet att kunna ta foto av egna handritade siffror och låta den framtagna maskininlärningsmodellen prediktera vilken siffra det är.

För att uppfylla syftena så kommer vi behöva besvara ett antal frågeställningar.

1.2 Frågeställning

Kan någon av de valda modellerna uppnå en ”accuracy score” på mer än 95 %?

Är någon av siffrorna svårare än övriga att prediktera?

1.3 Avgränsning

I uppgiften har inte modeller inom Neurala nätverk eller Deep learning vägts in.

2 Teori

Inom maskininlärning förekommer framför allt två huvudkategorier:

Väglett lärande (eng. Supervised learning) som innebär att modellen lär sig genom att den blir vägledd av en beroende variabel (eng. label) som är det faktiska svaret för varje enskild observation (Géron, 2019, p.8).

Väglett lärande kan delas in i:

- Regressionsproblem, där den beroende variabeln har kontinuerliga värden
- Klassificeringsproblem, där den beroende variabeln kan anta olika klasser.

Icke väglett lärande (eng. Unsupervised learning) som innebär att det inte finns någon beroende variabel (eng. label), vilket gör att modellen behöver lära sig utan vägledning (Géron, 2019, p.9).

Icke väglett lärande kan delas in i:

- Klustring, där modellen delar in data i olika grupper/segment.
- Dimensionsreducering, där modellen förenklar data genom att reducera antalet dimensioner.

Två andra kategorier som inte behandlas i denna rapport är *Semisupervised learning* och *Reinforcement learning*.

Den aktuella uppgiften är ett typiskt klassificeringsproblem, där den beroende variabeln är av en viss klass (i detta fall en siffra mellan 0 och 9). För att kunna ta itu med problemet har olika modeller för klassificeringsproblem valts ut för det aktuella datasetet.

2.1 Modeller

I denna rapport används maskininlärningsmodeller från scikit-learn. Scikit-learn är ett bibliotek av modeller och metoder inom maskininlärning (Streamlit, 2024).

2.1.1 Support Vector Machine

Support Vector Machine (förkortat SVM) är en model som kan användas både vid regressionsproblem och vid klassificeringsproblem. SVM är känslig för skalan på data vilket kan hanteras genom standardisering (Gerón, 2019, p.154). Vid klassificeringsproblem kan modellen Support Vector Classifier (förkortat SVC) användas.

Exempel på hyperparametrar är: kernel, gamma och C.

2.1.2 Random Forest Classifier

Random Forest kan också användas för både regressionsproblem och klassificeringsproblem. En Random Forest-modell beräknar ett givet antal Beslutsträd och ger den prediktion som är bäst i genomsnitt. Vid klassificeringsproblem kan modellen Random Forest Classifier användas (Gerón, 2019, p.197).

Exempel på hyperparametrar är: n_estimators, min_samples_split och min_samples_leaf.

2.1.3 Nearest Neighbor Classifier

En Nearest neighbor-metod går ut på att hitta ett, i förväg definierat, antal datapunkter närmast den nya punkten och därigenom prediktera det som eftersöks. Vid klassificeringsproblem kan modellen K-Nearest Neighbors Classifier användas (Gerón, 2019, p.106).

Exempel på hyperparametrar är: n_neighbors och weights

2.1.4 Voting Classifier

Med Voting Classifier kan vi prediktera med alla de tre föregående modellerna, där den prediktion som fått flest ”röster” (eng. hard voting) alternativt högst sannolikhet per klass (eng. soft voting) avgör (Gerón, 2019, pp.189-192).

Exempel på hyperparametrar är: voting och weights

2.2 Preprocessing

2.2.1 Standard Scaler

Med Standard Scaler standardiseras data så att medelvärdet blir 0 och standardavvikelsen 1. Det är ett nödvändigt moment för flera modeller för att de ska fungera optimalt (Gerón, 2019, p.69).

2.2.2 Principal Component Analysis

Principal Component Analysis (förkortat PCA) är en modell för dimensionsreducering. PCA kan till exempel användas tillsammans med SVM-modeller för att minska beräkningstiden, utan att märkbart påverka träffsäkerheten Gerón, 2019, p.213).

2.3 Utvärderingsverktyg

2.3.1 Confusion Matrix

Ett bra sätt att utvärdera en klassificeringsmodell på är med hjälp av en *Confusion Matrix*. Raderna representerar de sanna värdena/klasserna och kolumnerna representerar de predikterade värdena/klasserna (Gerón, 2019, p.91). Om en klassificerare är 100 % korrekt kommer *False Negative* och *False Postive*, i tabellen till höger, få värdet 0.

En matris som kan användas för att utvärdera hur bra modellen predikterar respektive klass.

Confusion Matrix		
	Predicted Negative	**Predicted Positive**
Actual Negative	True Negative	False Positive
Actual Positive	False Negative	True Positive

Tabell 1. Principiell bild av Confusion Matrix

2.3.2 Recall

Recall mäter andelen korrekta prediktioner av den sanna positiva klassen (nedersta raden i tabell 1), och beräknas med formeln:

$$\text{Recall} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Negative}}$$

2.3.3 Precision

Precision mäter andelen korrekta prediktioner av de positiva prediktionerna (högra kolumnen i tabell 1), och beräknas med fomeln:

$$\text{Precision} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Positive}}$$

2.3.4 F1

F1 är en kombination av Recall och Precision och ger en mer samlad bedömning. En hög F1-poäng erhålls enbart om både Recall och Precision är höga (Gerón, 2019, p.92), och beräknas med formeln:

$$F_1 = \frac{2}{\left(\frac{1}{\text{Precision}} + \frac{1}{\text{Recall}}\right)} = \frac{\text{True Positive}}{\text{True Positive} + \frac{(\text{False Negative} + \text{False Positive})}{2}}$$

2.3.5 Accuracy

Accuracy går bra att använda om datasetet är balanserat och inte har någon/några klasser som förekommer betydligt oftare än andra. Accuracy kan beräknas från resultatet av en Confusion Matrix, genom att beräkna andelen korrekta prediktioner av alla genomförda prediktioner:

$$\text{Accuracy} = \frac{\text{True Positive} + \text{True Negative}}{\text{True Positive} + \text{False Positive} + \text{True Negative} + \text{False Negative}}$$

2.4 Streamlit

Streamlit är ett ramverk för att skapa applikationer för maskininlärning med hjälp av Python. Det möjliggör för fler användare då det inte kräver kunskaper om att bygga web-applikationer (Streamlit, 2024).

3 Metod

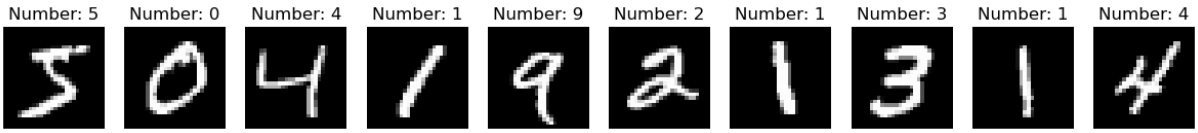
3.1 Modellen

Data har inhämtats från MNIST-databasen och består av data för 70 000 bilder på siffror (0–9), varav 60 000 avser träningsdata och 10 000 avser testdata.

Datasetet består av 70 000 samples och har delats in i träningsdata, valideringsdata och testdata. Träningsdatasetet består av 50 000 samples, valideringsdatasetet är 10 000 samples och testdatasetet är 10 000 samples.

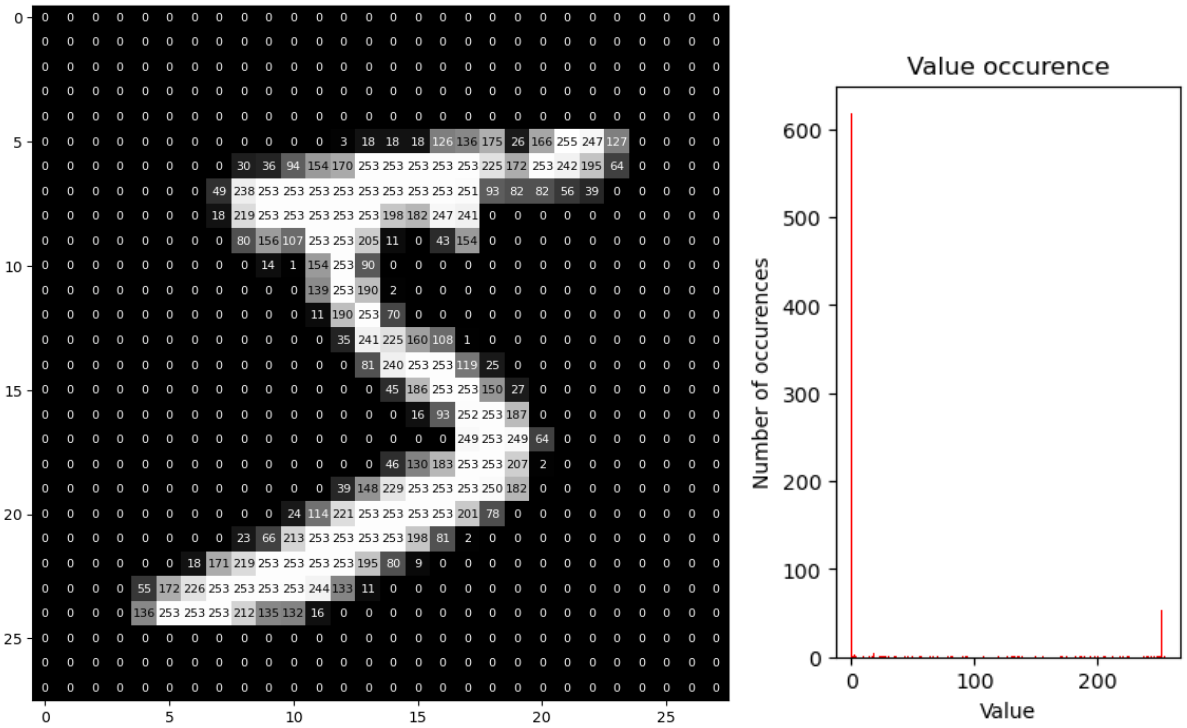
3.1.1 Utforska och förstå inhämtade data

Varje bild representeras av data bestående av 784 kolumner (features), som var och en motsvarar varje pixel i bilden, som är 28x28 (= 784) pixlar stor. Figuren nedan visar exempel på handritade siffror som förekommer i träningsdatasetet.



Figur 1. Exempel på siffror som förekommer i träningsdatasetet.

Varje pixel har ett värde (mellan 0 – 255) som utgör färgintensiteten, där 0 representerar svart och 255 representerar vitt, se figur nedan.

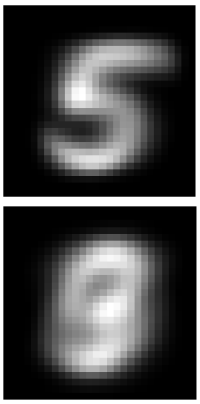


Figur 2. T.v. Handritad siffra (5) med värdet för respektive pixel. T.h. Histogram över fördelningen av respektive pixelvärde.

Som vi kan se i histogrammet ovan har majoriteten av pixlarna värdet 0 (vilket utgör den svarta bakgrunden) och själva siffran (vitfärgad i Figur 2) utgörs av värden nära 255.

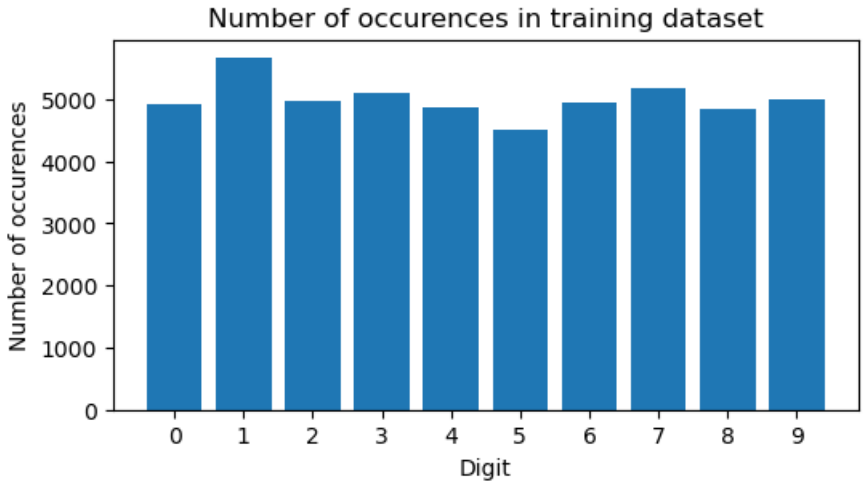
Genom att visa en bild med pixlarnas medelvärden för siffran 5 i träningsdatasetet, kan vi se att det fortfarande liknar siffran 5, om än suddig. Detta indikerar att siffran 5 har i genomsnitt ungefär samma form och är centrerad i bild. Se den övre bilden i Figur 3.

Om vi i stället visar en bild med pixlarnas medelvärden för alla siffror i träningsdatasetet ser vi att de i genomsnitt är samlade i mitten och generellt sett är det en ”tom” ram runt. Se den nedre bilden i Figur 3.



Figur 3. Pixlars medelvärden

Genom att kontrollera förekomsten av respektive siffra (0–9) bedöms träningsdatasetet vara någorlunda balanserat, se figur nedan.



Figur 4. Fördelning av hur många gånger varje siffra förekommer i träningsdatasetet.

3.1.2 Val av modeller

Det är ett typiskt klassificeringsproblem att prediktera om siffran tillhör någon av klasserna 0–9. Därmed har de modeller som ingår i denna rapport valts för att de kan användas vid klassificeringsproblem.

3.1.2.1 Val av hyperparametrar

Genom att pröva och jämföra modeller med olika hyperparmetervärden kan den bästa modellen identifieras och väljas ut. Processen kan vara tidskrävande om många olika hyperparametrar prövas. Därför tränas modellerna på ett mindre träningsdataset (i detta fall 10 000 samples) och utvärderas på ett mindre valideringsdataset (i detta fall 2 000 samples).

För att snabba på beräkningshastigheten ytterligare används PCA för den data som SVC-modellen tränar på.

Om det är okänt vilka hyperparametervärden som lämpar sig för det aktuella datasetet kan det första steget vara att börja med ett större intervall för att få en grov uppskattning. Därefter smalnas intervallet av mer och mer tills ett tillfredställande resultat uppnås.

3.1.2.2 Träna optimerade modeller och utvärdera på valideringsdata

Varje modell med utvalda hyperparametrar tränas sedan på hela träningsdatasetet (50 000 samples) och utvärderas mot valideringsdatasetet (10 000 samples). På detta sätt blir de olika klassificerarna mer jämförbara, eftersom storleken på träningsdatasetet kan spela roll för resultatet.

Här utvärderas modellerna på en samlad bedömning utifrån poäng (”accuracy”, ”recall”, ”precision” och ”F1-score”) samt om enstaka siffror är uppenbarligen svåra, enligt Confusion Matrix.

3.1.2.3 Utvärdera bästa modellen på testdata

Till sist tränas den bästa modellen på utökade träningsdata (bestående av träningsdata och valideringsdata) och predikteras sedan på testdata (10 000 samples) för att se utfallet och jämföra poängen med den för valideringsdata, för att se så den inte ändrats för mycket.

Modellens resultat på testdata jämförs med resultatet på valideringsdata för att se att prediktionen håller samma höga nivå även här.

Den tränade modellen sparas även som en fil för att kunna användas i Streamlit-applikationen.

3.2 Streamlit-applikationen

Den stora utmaningen med denna del är att lista ut vad vi behöver göra med den bild som tas med kameran, för att den ska gå bra att prediktera med vår framtagna modell.

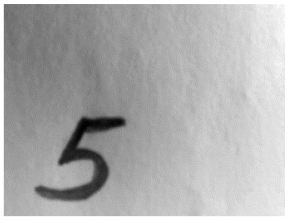
Tillvägagångssättet förutsätter att vi förstår både den data som modellen har tränats på och den data som genereras då kameran tar en bild. För att predikteringen ska fungera bra så behöver den bild som fås via kameran prepareras på olika sätt för att efterlikna siffrorna i träningsdatasetet.

För att ge ett bra resultat vid predikteringen bearbetas bilden genom följande moment:

3.2.1 Steg 1: gråskala

Bilden omvandlas från färg till gråskala med hjälp av scikit-image. Härigenom fås endast en färgdimension, gråskala.

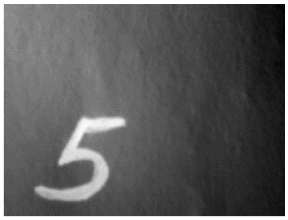
Datan normaliseras sedan på skalan 0–255 för att få bättre spridning inom gråskalan.



Figur 5. Omvandlad till gråskala

3.2.2 Steg 2: inverterad gråskala

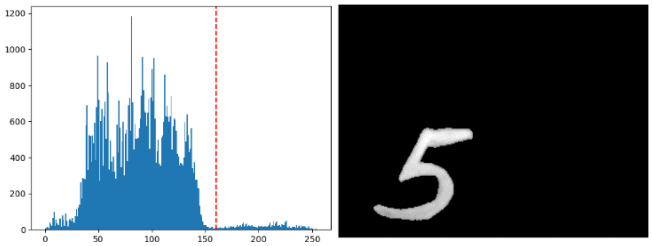
Gråskalan inverteras för att stämma överens med träningsdata (som har vit siffra på svart bakgrund). Det har utförts med hjälp av scikit-image, men går även att göra manuellt på formeln ”nytt värde” = 255 - ”gamla värdet” för varje pixel.



Figur 6. Invertering av gråskala

3.2.3 Steg 3: homogen bakgrund med värde 0

Då pappret (bakgrunden) kan bli grå och få nyansskillnader då bilden tas, sätter vi alla värden under en viss nivå till värde 0 (i syfte att hela bakgrunden ska bli svart). Här är tröskelvärdet satt till 160 (på skalan 0–255).

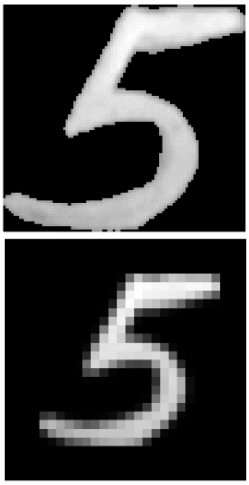


Figur 7. Svart bakgrund genom att sätta lägre gråskalevärden till 0.

3.2.4 Steg 4: Skala om och centrera till 28x28-format

För att den skrivna siffran kan ha en annan storlek eller fått en annan placering i bild jämfört med hur det ser ut i träningsdatasetet så skalas siffran om och centreras. Detta utförs manuellt med genom att skala bort de rader och kolumner som bara innehåller svart (värde 0) i varje pixel, dvs de skär inte genom den vita siffran. Kvar blir ett urklipp av siffran, se bild ovan till höger.

För att kunna prediktera siffran behöver vi skala om bilden till 28x28 pixlar och gärna centrera siffran i bilden så att den efterliknar träningsdatasetet. Som vi såg tidigare när vi undersökte träningsdatasetet (kap 3.1.1) är det generellt sett en svart ram runt siffran. Den urklippta bilden skalas därför om till 20 pixlars höjd och centreras i en svart 28x28 bild, med en ram på 4 pixlar.



Figur 8. Urklipp (ovan) och omskalad (nedan).

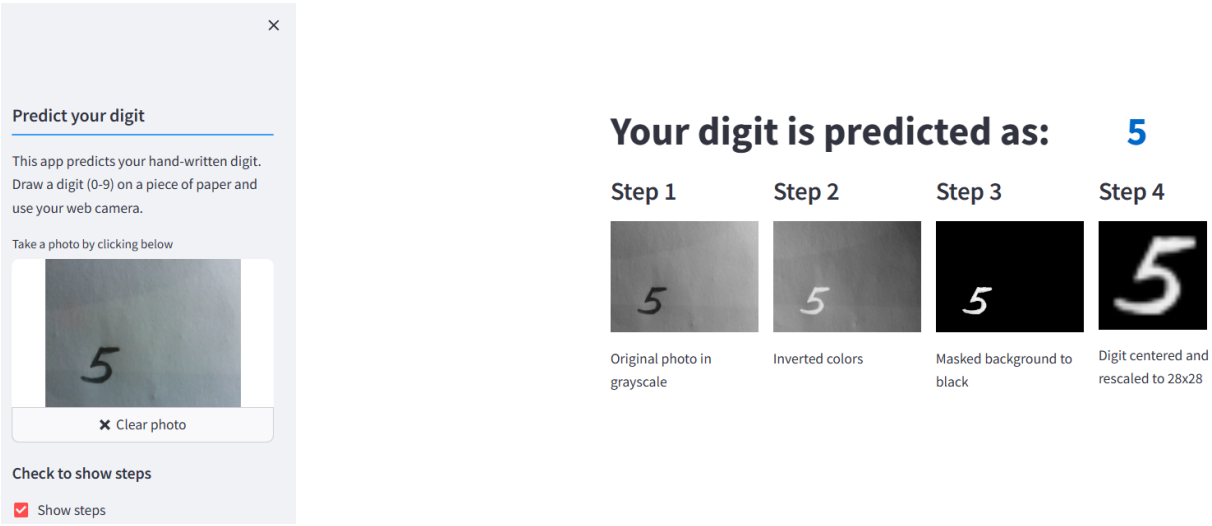
3.2.5 Steg 5: Standardisering och prediktering

Eftersom modellen har tränats på standardiserade data så behöver vi på motsvarande sätt standardisera data för den handritade siffran. Slutligen predikteras siffran med hjälp av den tränade modellen.

3.2.6 För användaren

Applikationen förutsätter att siffran skrivs med mörk penna på ljust papper för att transformeringen ska fungera som det är tänkt.

I applikationen kan användaren kryssa i en ruta för att se de olika stegen som genomförs för att bearbeta bilden innan den predikteras. Denna funktion skapades för att öka förståelsen och underlätta bildbearbetningsarbetet, men kan även ge användaren inblick i processen.



Figur 9. Exempel på bild från datorkameran som bearbetas och förbereds i Streamlit-applikationen innan prediktering.

4 Resultat och Diskussion

4.1 Val av hyperparameterar

Av de hyperparametrar som testades framkom följande resultat för bästa modell:

- **SVC**: SVC(C=0.5, gamma=1, kernel='poly'), alla testade parametervärden på *gamma* och *C* fick samma score, men för *kernel* var *poly* bättre än *rbf*.
- **Random Forest**: RandomForestClassifier(max_depth=10, min_samples_split=2).
- **KNN**: KNeighborsClassifier(n_neighbors=4, weights='distance').

4.2 Utvärdering på valideringsdatasetet

Efter att modellerna tränats på träningsdatasetet (med 50 000 samples) och validerats mot 10 000 samples, gav det följande resultat:

Resultat för olika modeller				
Modell	**F1-score**	**Recall**	**Precision**	**Accuracy**
SVC	Min: 0.97 Max: 1.00	Min: 0.96 Max: 1.00	Min: 0.98 Max: 1.00	0.98
Random Forest	Min: 0.90 Max: 0.98	Min: 0.92 Max: 0.98	Min: 0.94 Max: 0.98	0.95
KNN	Min: 0.93 Max: 0.98	Min: 0.91 Max: 0.99	Min: 0.92 Max: 0.97	0.95
Voting Classifier	Min: 0.95 Max: 0.99	Min: 0.95 Max: 0.99	Min: 0.96 Max: 0.99	0.97

Tabell 2. Poängbedömning för de valda modellerna.

Utifrån resultatet i tabellen ovan utsågs den bästa modellen, vilket blev SVC-modellen.

Lägst Recall-poäng hade SVC-modellen för siffan 9 som därmed var den svåraste siffran att prediktera korrekt.

Lägst Precision-poäng hade SVC-modellen för siffran 8 som modellen därmed felaktigt predikterat andra siffror som.

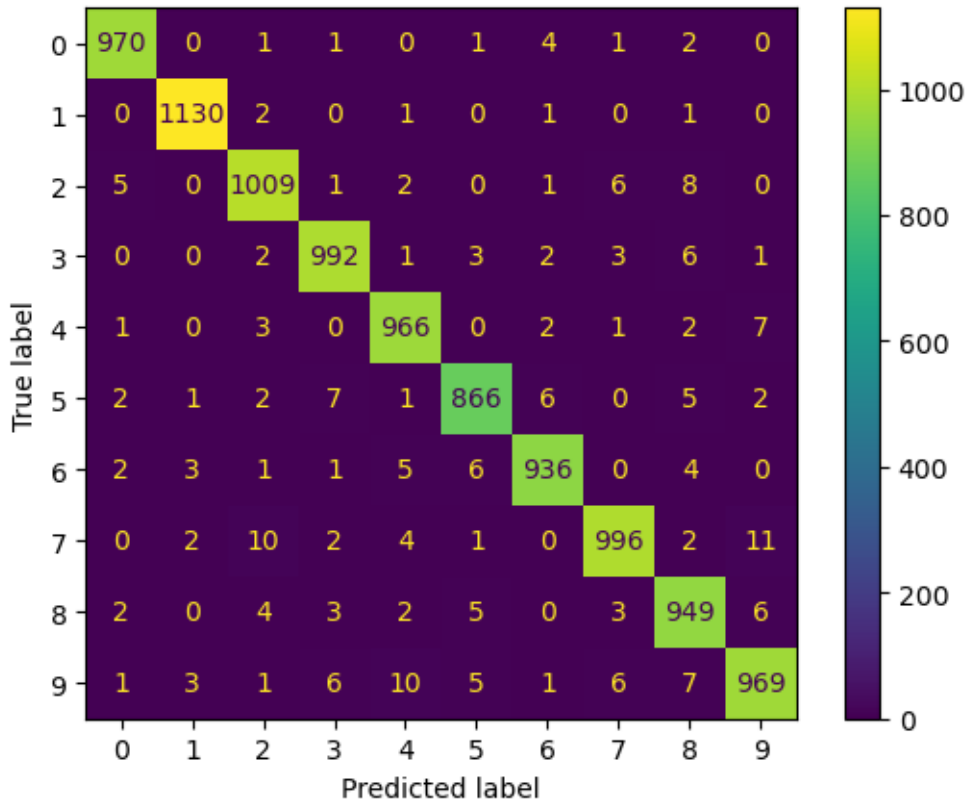
4.3 Utvärdering på testdatasetet

Efter att SVC-modellen tränats om på en sammanslagning av den tidigare träningsdatan och valideringsdatan, och predikterat testdata blev resultaten följande:

- F1-score: Min: 0.97 Max: 0.99
- Recall: Min: 0.96 Max: 1.00
- Precision: Min: 0.96 Max: 0.99
- Accuracy: 0.98

Att poängen inte märkbart förändrats jämfört med resultatet för valideringsdatan tyder på att modellen varken är över- eller underanpassad.

Den siffra som predikterades fel flest gånger var siffra 9, se figur nedan. Den predikterades fel 40 gånger, vilket utgör ca 4 % av alla korrekta 9:or. Se även Appendix A för utseendet på de 9:or som modellen inte predikterade rätt.



Figur 10. Confusion matrix för den slutliga modellen

4.4 Streamlit-applikationen

Streamlit-applikationen predikterar för det mesta korrekt. Det kan uppstå svårigheter vid prediktion om siffran som fotograferas är dåligt belyst. Då blir bakgrunden (pappret) alltför mörk vilket kan göra att det blir svårt att särskilja bakgrund från siffra.

Siffran får heller inte skilja sig märkbart från det genomsnittliga utseendet av en siffra i träningsdatasetet.

5 Slutsatser

Här besvaras frågeställningarna som ställdes i kapitel 1.2.

Kan någon av de valda modellerna uppnå ett ”accuracy score” på mer än 95 %?

Den bästa modellen, som var SVC, uppnådde ett ”accuracy score” på ca 98 %.

Är någon av siffrorna svårare än övriga att prediktera?

Modellen hade svårast att prediktera siffran 9. Andra siffror som var bland de svårare är 7 och 5.

Modellen presterar inte bättre än den data den tränas på. Om modellen ska kunna känna igen fler varianter av siffrorna (t.ex både 4 och 4 eller 7 och 7) så behöver den tränas på ytterligare träningsdata som innehåller de varianterna.

Beroende på vad som är viktigast av poäng eller beräkningstid och om ett visst mått är viktigare än ett annat kan ett annat val av modell göras. I detta fall var beräkningstiden för SVC-modellen visserligen längre än för övriga modeller men samtidigt inte för lång för att uteslutas.

Streamlit-applikationen är känslig för ljusförhållanden då den handritade siffran fotograferas. Den handritade siffran får heller inte skilja sig märkbart från det genomsnittliga utseendet på siffra som modellen tränats på.

5.1 Vidare arbete

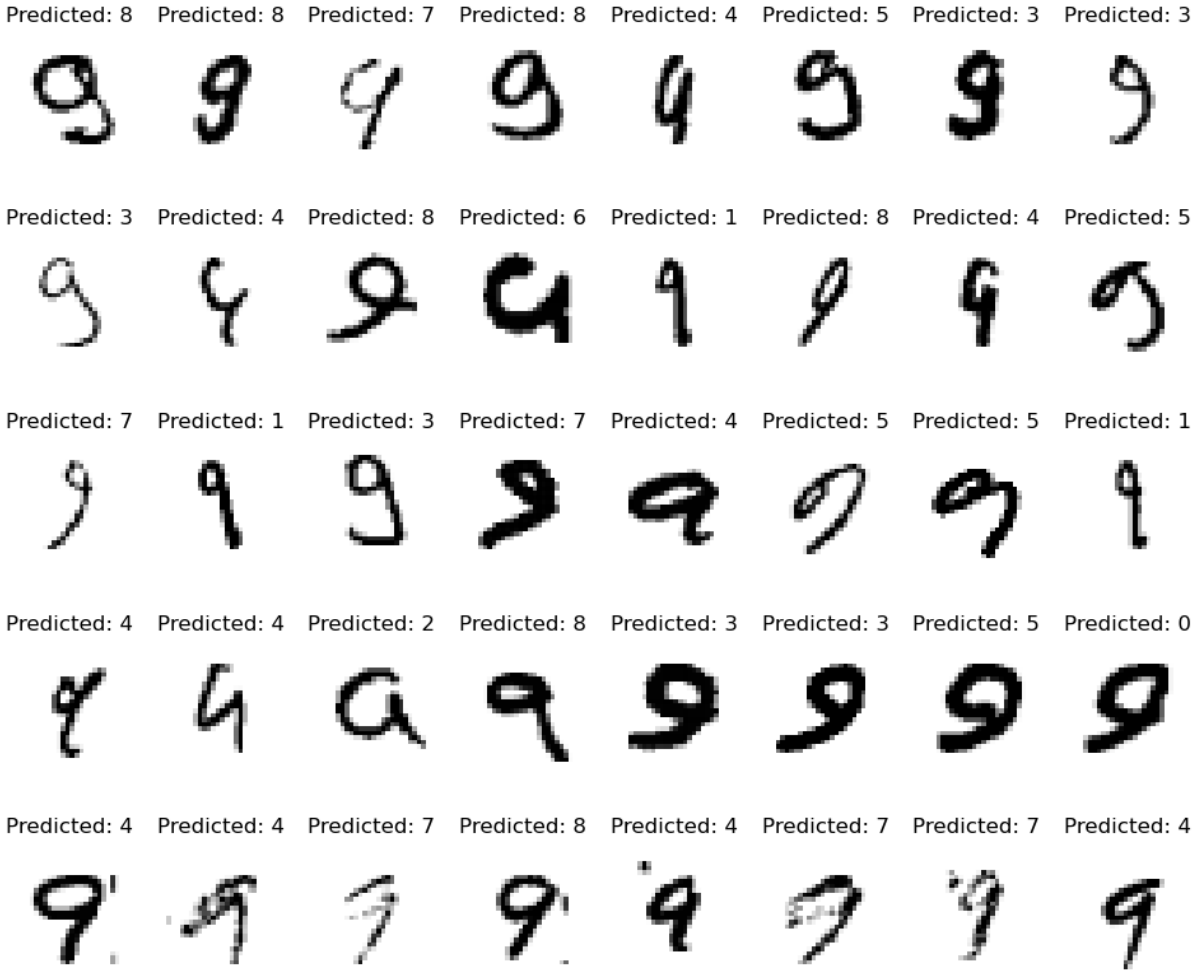
Det går att utvärdera fler modeller med andra val av estimatorer och andra val av hyperparameterar, för att se om det går att optimera ytterligare.

I denna uppgift har inte modeller inom neurala nätverk och deep learning använts, vilka kanske hade kunnat åstadkomma ännu bättre resultat.

Det finns möjlighet att göra ännu mer förbearbetning (preprocessing) av den fotograferade bilden för att täcka in fler aspekter, till exempel om det inte är en mörk siffra ritad på ljust papper.

Appendix A

Utseendet på siffrorna 9 som predikterades fel av den slutliga modellen:



Källförteckning

Géron, A. (2019). *Hands-on Machine Learning with Scikit-Learn, Keras & Tensorflow: Concepts, Tools and Techniques to Build Intelligent Systems (2nd ed.).* Sebastopol: O'Reilly Media, Inc.

Scikit-learn. (2024). Hämtat från https://scikit-learn.org/stable/modules/classes.html

Streamlit. (2024). Hämtat från https://streamlit.io